INTERNET

ราชบดินทร์ สุวรรณคัณฑิ

พื้นฐานการเขียน CGI PROGRAMMING ด้วย PERL 5 : เริ่มต้นการเขียนโปรแกรม PERL

รู้จักหน้าตาและส่วนประกอบเบื้องต้นของ PERL รวมทั้งตัวอย่างการเขียนโปรแกรมแบบง่าย ๆ

สวัสดีครับ กลับมาพบกันเป็นครั้งที่สองแล้วนะครับสำหรับเรื่องราวของการเขียนโปรแกรมบนระบบเครือข่ายอินเทอร์เน็ต ที่เราเรียกกันว่า CGI Programming โดยการใช้ภาษาที่เราเรียกกันว่า PERL ซึ่งในครั้งที่แล้วผมได้แนะนำให้คุณรู้จักเรื่องราว ลักษณะและประโยชน์ของการเขียน CGI Programming ไปแล้ว และผมยังได้กล่าวถึงชนิดของโปรแกรมที่เราจะนำมาใช้ในการเขียน CGI ซึ่งนับตั้งแต่ครั้งนี้เป็นต้นไปผมจะกล่าวถึง รูปแบบและรายละเอียดของการเขียนโปรแกรม CGI โดยการใช้ PERL ซึ่งการเขียนโปรแกรม PERL นั้นเราสามารถทำได้ทั้งบนระบบปฏิบัติการยูนิกซ์ และระบบปฏิบัติการ Windows NT แต่ในบทความชุดนี้จะเป็นการกล่าวถึงการเขียนโปรแกรมบนระบบปฏิบัติการยูนิกซ์ เนื่องจากการเขียน PERL บนยูนิกซ์นั้นได้รับความนิยมเป็นอย่างมากในปัจจุบัน ส่วนผู้ใช้ท่านใดที่อยากเขียน PERL บน Windows NT ก็สามารถทำได้เหมือน ๆ กับการเขียนโปรแกรมบนยูนิกซ์ ซึ่งผมขออนุญาตที่จะไม่กล่าวถึงเรื่องราวของพื้นฐานการใช้งานระบบยูนิกซ์นะครับ ดังนั้นถ้าคุณเป็นคนหนึ่งที่สนใจในเรื่องราวของการเขียน CGI Programming ด้วย PERL แต่ยังไม่มีความรู้เรื่องของระบบยูนิกซ์ อาจจะต้องไปหาหนังสือเกี่ยวกับระบบยูนิกซ์ที่มีวางขายอยู่ทั่วไปตามร้านหนังสือคอมพิวเตอร์มาอ่าน เพื่อเป็นการปูความรู้พื้นฐานก่อนนะครับ ก็เพียงแค่ขอให้รู้จักธรรมชาติของการทำงานบนระบบยูนิกซ์ และสามารถใช้คำสั่งพื้นฐานได้ก็น่าจะเพียงพอแล้ว

อะไรคือ PERL ?

PERL ย่อมาจากคำว่า Practical Extraction and Report Language คือภาษาที่ใช้ในการเขียนโปรแกรมคอมพิวเตอร์ภาษาหนึ่งที่เราเรียกกันว่าเป็น Interpreted Language มีลักษณะหรือความสามารถพิเศษในเรื่องของการจัดการกับเท็กซ์ไฟล์ เช่น การตรวจสอบและคัดเลือกข้อมูลที่เราสนใจจากเท็กซ์ไฟล์ที่มีอยู่ในระบบ นอกจากนั้น PERL ยังมีความสามารถในเรื่องของการปฏิบัติการกับ report เช่น เรื่องราวของการ printing report ที่มีรูปแบบแตกต่างกันออกไปตามข้อมูลที่เราสนใจ ซึ่ง PERL นับว่าเป็นภาษาโปรแกรมที่เหมาะสมกับงานทางด้านการจัดการระบบ (System Management) ซึ่งภาษานี้ถูกออกแบบมาเพื่อให้เป็นภาษาสำหรับการฝึกหัดเขียนโปรแกรม ดังนั้น PERL จะเป็นภาษาที่ง่ายต่อการเรียนรู้ง่ายต่อการเขียนและนำไปใช้ อีกทั้งยังเป็นภาษาที่มีประสิทธิภาพในการทำงานที่เข้าขั้นดีทีเดียว

นอกจากนั้น PERL นับว่าเป็นภาษาที่เป็นการรวมเอาคุณสมบัติเด่น ๆ ของภาษา C, sed, awk และ sh เอาไว้ด้วยกัน ดังนั้นผู้อ่านท่านใดที่เคยเขียนโปรแกรมที่ผมว่านี้มาก่อน คุณจะสามารถเข้าใจในเรื่องของการเขียนโปรแกรม PERL ได้อย่างสบายเลย เพราะกฎเกณฑ์ต่าง ๆ

ที่ใช้ในการเขียนโปรแกรมจะมีลักษณะที่คล้ายคลึงกับการเขียนโปรแกรมในภาษา C ตรงนี้ผมมีประสบการณ์มาแล้ว คือว่าที่ทำงานผมมีพี่คนหนึ่งซึ่งเคยเขียนภาษา C มาแต่ยังไม่เคยเขียน PERL เลย แต่พอพี่คนนี้มาลองเขียน PERL ดูก็ปรากฏว่าใช้เวลาในการเรียนรู้แค่อาทิตย์เดียวเองครับ

เตรียมพร้อมก่อนเริ่มเขียนโปรแกรม

หลังจากที่เราได้รู้จักกับ PERL กันไปพอหอมปากหอมคอแล้ว คำถามต่อไปที่หลาย ๆ คนอยากทราบก็คือ การเขียนโปรแกรมด้วย PERL ต้องมีเครื่องไม้เครื่องมือหรืออุปกรณ์อะไรบ้าง ครับ... ในหัวข้อนี้จะกล่าวถึงเรื่องราวของอุปกรณ์ นั่นหมายถึงซอฟต์แวร์ที่ต้องมี และหมายความรวมไปถึงความรู้พื้นฐานที่คุณควรจะมี ก่อนที่เราจะเริ่มต้นการเขียนโปรแกรม PERL กัน

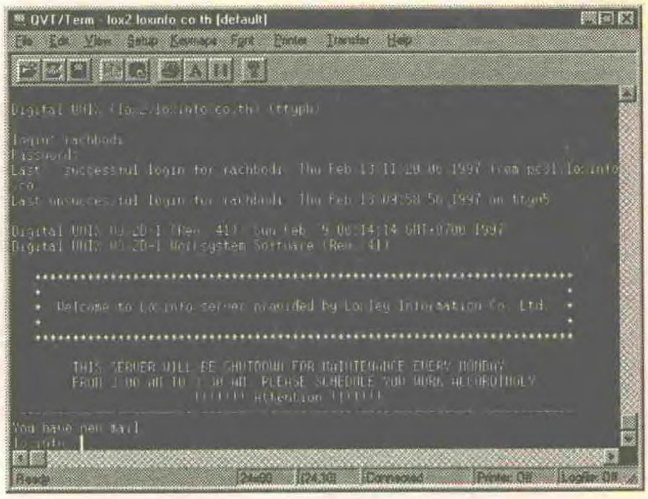
1. ระบบปฏิบัติการ ในการทำงานของ PERL นั้นสามารถทำงานได้บนระบบปฏิบัติการยูนิกซ์หรือระบบปฏิบัติการ Windows NT นั่นหมายความว่าคุณจะต้องมีระบบปฏิบัติการใดระบบปฏิบัติการหนึ่งในสองระบบปฏิบัติการที่ผมกล่าวมานี้ ซึ่งส่วนใหญ่เรามักจะนิยมการเขียน PERL บนระบบปฏิบัติการยูนิกซ์กันซะมากกว่า

ทีนี้หลายคนก็เริ่มสงสัยต่อไปอีกว่า แล้วเราจะเอาระบบปฏิบัติยูนิกซ์จากที่ไหนมาใช้ อันนี้ไม่ยากเลยครับ มีทางให้เลือกสองทาง ทางแรกก็คือ การเป็นสมาชิกของศูนย์บริการแห่งใดแห่งหนึ่ง ไม่ว่าจะเป็นบริษัทที่เป็น ISP (Internet Service Provider) ที่มีอยู่มากมายในบ้านเราหรือไม่ก็ตามสถาบันการศึกษาต่าง ๆ ที่มักจะมีตัว Server ที่มีระบบปฏิบัติการยูนิกซ์เป็นระบบปฏิบัติการหลักบนเครื่อง แถมในบางที่ที่ผมกล่าวมานี้อาจมีโปรแกรมคอมพายเลอร์ของ PERL ให้คุณได้ใช้งานด้วย ส่วนอีกวิธีหนึ่งก็คือ คุณหาโปรแกรมระบบปฏิบัติการยูนิกซ์มาใช้ที่บ้านเองเลย ซึ่งในปัจจุบันนี้มีระบบปฏิบัติการยูนิกซ์ที่สามารถทำงานได้บนเครื่อง PC ทั่ว ๆ ไปให้คุณได้เลือกใช้กันอยู่มากมาย เช่น ระบบยูนิกซ์ที่ผมใช้งานบนเครื่อง PC ที่บ้านอยู่ก็คือ FREE BSD UNIX หรือบางคนอาจจะใช้ BSDI UNIX ก็ได้นะครับ ซึ่ง FREE BSD ที่ผมใช้อยู่นั้นเป็นเวอร์ชัน 2.1.6 สั่งซื้อผ่านทางอินเทอร์เน็ตเลยนะครับเนี่ย ราคาก็ประมาณ 800 กว่าบาทถึงพันกว่าบาทต้น ๆ นิดหน่อยครับ ถ้าใครสนใจก็ลองเข้าไปดูได้ที่ไซท์ของ Walnut Creek CDROM ที่

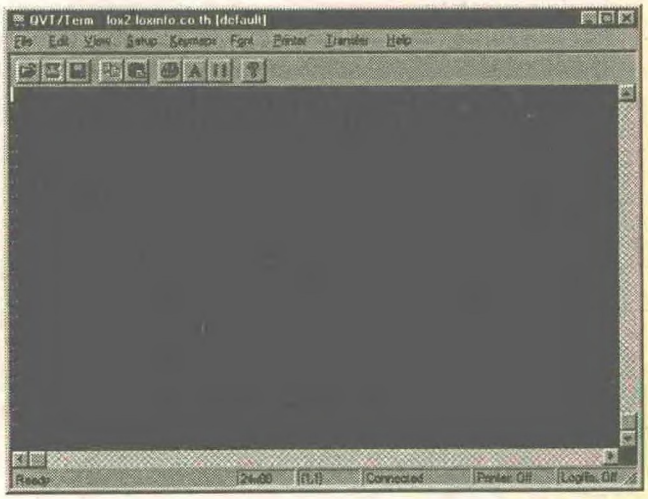
```
http://www.cdrom.com
```

2. โปรแกรมคอมพายเลอร์หรือตัวแปรภาษาของ PERL อันนี้เราสามารถดาวน์โหลดมาใช้ได้ฟรีจากอินเทอร์เน็ต ซึ่งคอมพายเลอร์ของ PERL จะมีทั้งแบบที่ทำงานบนระบบยูนิกซ์ และแบบที่ทำงานบนระบบปฏิบัติการ Windows NT และในปัจจุบันได้มีคอมพายเลอร์ของ PERL เวอร์ชัน 5.003 ที่ทำงานบนระบบปฏิบัติการยูนิกซ์ให้ดาวน์โหลดได้ที่

```
http://www.perl.com/perl/info/software.html
```



รูปที่ 1 หน้าจอของโปรแกรม telnet เมื่อเรา connect เข้าสู่ระบบยูนิกซ์



รูปที่ 2 หน้าตาของโปรแกรม VI เอดิเตอร์ยอดนิยมในยูนิกซ์

3. โปรแกรมเท็กซ์เอดิเตอร์ตัวไหนก็ได้ ซึ่งผมเห็นหลาย ๆ ท่านที่ใช้ยูนิกซ์บ่อย ๆ มักจะใช้โปรแกรม VI ซึ่งเป็นโปรแกรมเอดิเตอร์บนยูนิกซ์ เพราะมีประสิทธิภาพในการจัดการกับข้อความที่เราพิมพ์เข้าไปได้ดีกว่าเท็กซ์เอดิเตอร์ทั่ว ๆ ไปที่เราใช้กันอยู่ อันนี้ก็แล้วแต่สะดวกนะครับ

4. เนื่องจากส่วนใหญ่การเขียน PERL มักจะนิยมเขียนกันบนระบบปฏิบัติการยูนิกซ์ ดังนั้นคุณควรจะมีความรู้ในเรื่องของการใช้งานยูนิกซ์มาบ้างนะครับ ไม่ว่าจะเป็นเรื่องของคำสั่งพื้นฐานต่าง ๆ รูปแบบโครงสร้างการเก็บข้อมูลในระบบยูนิกซ์ ตลอดจนชนิดของไดเรคทอรีต่าง ๆ ในระบบยูนิกซ์ ซึ่งความรู้เหล่านี้คุณสามารถหาอ่านได้จากหนังสือที่มีเรื่องราวของระบบยูนิกซ์ที่มีวางขายอยู่ทั่วไปครับ

โครงสร้างพื้นฐานในการเขียนโปรแกรม

การเขียนโปรแกรม PERL มีลักษณะการเขียนและโครงสร้างของตัวโปรแกรมคล้ายกันกับการเขียนโปรแกรมในภาษาซีหรือการเขียนโปรแกรมที่เป็นโครงสร้าง (Structure Programming) อื่น ๆ นั่นคือ โปรแกรม PERL จะมีโครงสร้างพื้นฐานดังนี้

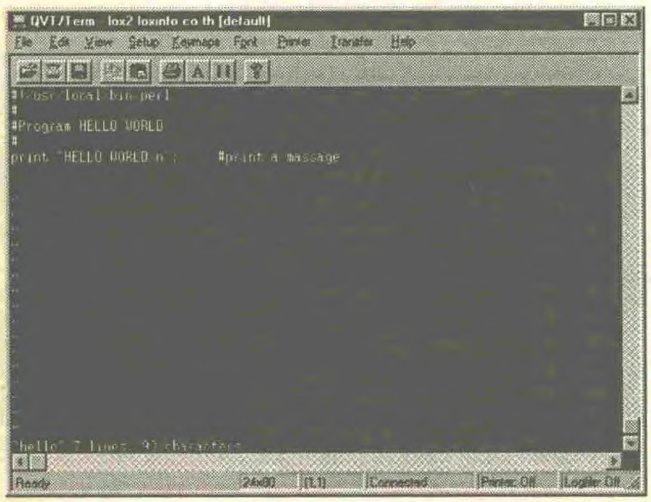
- HEADING PROGRAM
- DECLARATION SECTION
- BODY PROGRAM

- ส่วนของ HEADING PROGRAM หรือที่เราเรียกว่าส่วนหัวของโปรแกรมนั้น จะเป็นส่วนที่บอกถึงจุดเริ่มต้นของการเขียนโปรแกรม และยังเป็นส่วนที่บอกถึงไดเรคทอรีที่มีคอมพายเลอร์ของ PERL อยู่ หรือเป็นการบอกว่าจะให้โปรแกรมไปหาตัวคอมพายเลอร์ได้จากที่ไหน เช่น

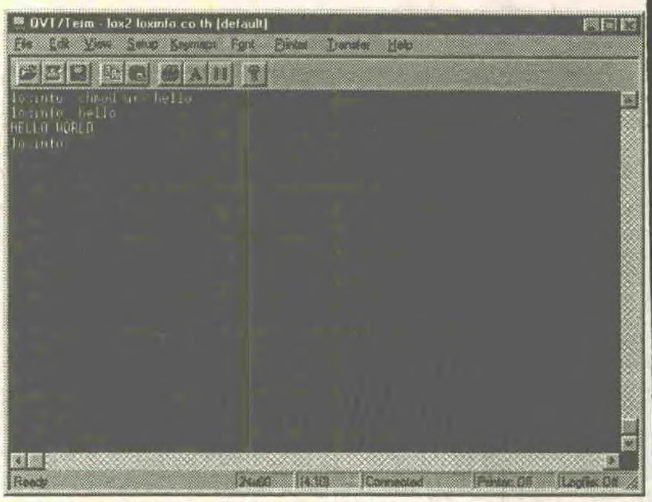
```
#!/usr/local/bin/perl
```

จากตัวอย่างที่ผ่านมาเป็นการกำหนดส่วนของ HEADING PROGRAM ซึ่งจะเป็นการบอกว่าถ้ามีการเรียกโปรแกรมนี้ทำงาน ให้ระบบไปหาตัวคอมพายเลอร์ของ PERL ได้จากไดเรคทอรี /usr/var/local/bin/perl ซึ่งไดเรคทอรีที่ว่านี้ก็จะแตกต่างกันไปตามการกำหนด Config ของยูนิกซ์ที่คุณใช้อยู่ แต่มีข้อควรสังเกตอยู่นิดหนึ่งก็คือ ส่วนใหญ่แล้วที่อยู่ของ PERL มักจะอยู่ภายใต้ไดเรคทอรี bin ของระบบยูนิกซ์ (อันนี้ถ้าใครใช้ยูนิกซ์ประจำคงจะทราบเหตุผลนะครับ)

- ส่วนของ DECLARATION SECTION คือส่วนที่ใช้ในการประกาศตัวแปร ว่าเราจะใช้ตัวแปรชื่ออะไร และมีชนิดของตัวแปรเป็นอะไร



รูปที่ 3 หน้าจอของ VI เมื่อเราพิมพ์ซอร์สโค้ดเสร็จแล้ว



รูปที่ 4 หน้าจอของยูนิกซ์เมื่อเราทำการรันโปรแกรม

- BODY PROGRAM คือส่วนที่เป็นที่รวมของคำสั่งที่ใช้ในการสั่งงาน หรือกำหนดรูปแบบของการทำงานของโปรแกรมที่เราเขียนขึ้นมา

เพื่อให้คุณได้เห็นภาพของการเขียนโปรแกรม PERL ได้ชัดเจนมากขึ้น ต่อไปนี้จะเป็นการแสดงตัวอย่างของการเขียนโปรแกรม PERL พื้นฐาน ที่เป็นโปรแกรมยอดฮิต นั่นคือโปรแกรม HELLO WORLD

```
#!/usr/local/bin/perl
#
# Program HELLO WORLD
#
print "HELLO WORLD\n";    # Print a massage
```

- บรรทัดแรกเป็นส่วนของ HEADING PROGRAM ซึ่งกฎเกณฑ์ในการเขียน HEADING PROGRAM คือจะต้องขึ้นต้นด้วยเครื่องหมาย #! แล้วถึงจะตามด้วย PATH ที่บอกถึงที่อยู่ของคอมพายเลอร์ของ PERL

- บรรทัดที่ 2, 3 และ 4 เป็นการแสดงให้คุณเห็นถึงการใช้งานเครื่องหมาย Comment ในการเขียน PERL ซึ่งการเขียน Comment นั้นทำได้โดยพิมพ์เครื่องหมาย # นำหน้าส่วนที่เป็น Comment ซึ่งข้อความทุกอย่างที่อยู่ถัดจากเครื่องหมาย Comment จะไม่มีการประมวลผลในขั้นตอนของการรันโปรแกรม และที่น่าสังเกตก็คือ ถ้าในกรณีที่คุณต้องการเขียน Comment หลาย ๆ บรรทัด คุณจะต้องขึ้นต้นแต่ละบรรทัดด้วยเครื่องหมาย Comment และถ้าคุณต้องการเขียน Comment เพื่ออธิบายการทำงานของโค้ดในบรรทัดใดบรรทัดหนึ่งก็สามารถทำได้ดังตัวอย่างในบรรทัดที่ 5

- บรรทัดที่ 5 แสดงคำสั่งพื้นฐานของ PERL ก็คือคำสั่ง print ซึ่งคำสั่งนี้จะใช้ในการแสดงข้อมูลที่อยู่ถัดจากคำสั่งนั้นออกมาที่หน้าจอภาพ เช่นในตัวอย่าง เมื่อมีการรันโปรแกรมขึ้น จะมีข้อความ HELLO WORLD ที่หน้าจอภาพ ส่วนเครื่องหมาย \n เป็นคำสั่งที่กำหนดให้ขึ้นบรรทัดใหม่เมื่อมีการแสดงข้อความเสร็จแล้ว

- ในการบอกถึงจุดสิ้นสุดในแต่ละคำสั่งนั้นเราจะใช้เครื่องหมายเซมิโคลอน (;) เป็นตัวคั่นเหมือนกับในภาษา PASCAL ซึ่งเครื่องหมายนี้จะใช้ในการบอกจุดสิ้นสุดของคำสั่ง และใช้คั่นระหว่างตัวแปรแต่ละชนิดที่เราตั้งขึ้นมาในส่วนของ DECLARATION SECTION ซึ่งในตัวอย่างแรกนี้เรายังไม่มีการประกาศตัวแปรใด ๆ เลย

หลังจากที่เราพิมพ์โปรแกรมเสร็จเรียบร้อยแล้ว ให้เราทำการเซฟ และพิมพ์คำสั่งต่อไปนี้ที่เครื่องหมายพรอมต์ของยูนิกซ์

```
chmod u+x progname
```

ซึ่ง progname นั้นคือชื่อของไฟล์ที่คุณตั้งขึ้นมาในขั้นตอนของการเซฟโปรแกรม และหลังจากนั้นถ้าคุณต้องการที่จะทำการรันโปรแกรม ก็สามารถทำได้โดยการใช้คำสั่งใดคำสั่งหนึ่งต่อไปนี้

```
perl progname
./progname
progname
```

แต่ถ้าคุณรันโปรแกรมแล้วปรากฏว่ามีข้อความแสดงข้อผิดพลาดเกิดขึ้น หรือไม่มีการแสดงข้อความ HELLO WORLD ให้คุณใช้คำสั่ง perl -w progname ซึ่งจะเป็นการให้แสดงข้อความเตือนและแสดงข้อความช่วยเหลือเกี่ยวกับข้อผิดพลาดที่เกิดขึ้น หรือถ้าคุณต้องการรันโปรแกรมพร้อมกับการแก้ไขให้ใช้คำสั่ง perl -d progname

ตัวแปรพื้นฐาน

ในตัวอย่างที่ผ่านมาผมได้แสดงให้เห็นถึงโครงสร้างพื้นฐานของการเขียนโปรแกรม PERL ซึ่งคุณจะเห็นว่าในการเขียนโปรแกรม PERL นั้นมีโครงสร้างของการเขียนโปรแกรมเหมือนกับการเขียนโปรแกรมโดยการใช้ภาษาโปรแกรมมิงอื่น ๆ แต่ในตัวอย่างที่แล้วเป็นการเขียนโปรแกรมให้ทำงานง่าย ๆ คือการแสดงข้อมูลที่เราต้องการออกมาที่หน้าจอภาพ โดยการใช้ฟังก์ชัน print

เรื่องต่อไปที่ผมกำลังจะกล่าวถึงก็คือ เรื่องของตัวแปรพื้นฐานที่เราสามารถใช้ได้ในการเขียนโปรแกรม PERL เพราะการใช้งานตัวแปรเพื่อทำการเก็บข้อมูลตลอดจนการทำงานต่าง ๆ ที่เกี่ยวข้องกับตัวแปรนั้นถือว่าเป็นเรื่องสำคัญที่จะต้องมีในการเขียนโปรแกรม ซึ่งการเขียนโปรแกรมโดยใช้ PERL ก็เช่นเดียวกัน เราสามารถใช้ตัวแปรพื้นฐานได้สองชนิดคือ

1. SCALAR VARIABLES คือตัวแปรพื้นฐานที่สุดของ PERL ซึ่งตัวแปรชนิดนี้เราสามารถใช้ในการเก็บข้อมูลทั้งที่เป็นแบบตัวอักษรหรือชุดของตัวอักษรและตัวเลขก็ได้ ดังตัวอย่างต่อไปนี้

```
$value = 9;
$value = "high";
$value = "9"
$value = "0005"
```

ในบรรทัดแรกเป็นการกำหนดตัวแปรชื่อ $value ให้เป็นชนิดตัวเลขมีค่าเท่ากับ 9 ส่วนในบรรทัดที่ 2, 3 และ 4 เป็นการกำหนดตัวแปร $value ให้เป็นชุดของตัวอักษรหรือที่เราเรียกกันว่า string ซึ่งชุดของตัวอักษรนั้นสามารถประกอบขึ้นมาจากตัวอักษรหรือตัวเลขตั้งแต่หนึ่งตัวขึ้นไป ข้อสังเกตคือถ้าเราต้องการกำหนดค่าให้กับตัวแปรชนิด string จะต้องมีเครื่องหมายคำพูด (" ") คล่อมค่านั้น ๆ ไว้ด้วย ดังตัวอย่างในบรรทัดที่ 2, 3 และ 4

นอกจากนั้นในการกำหนดชื่อของตัวแปรชนิด SCALAR นั้นจะต้องขึ้นต้นด้วยเครื่องหมาย $ จากนั้นตามด้วยชื่อของตัวแปร ซึ่งประกอบขึ้นมาจากตัวอักษร ตัวเลข และเครื่องหมาย underscores (_) ซึ่งจะต้องมีตัวขึ้นต้นเป็นตัวอักษรเท่านั้น และในการกำหนดตัวแปรคุณจะต้องระวังเรื่องตัวใหญ่หรือตัวเล็กด้วย (case sensitive) เช่น ตัวแปร $a จะถือว่าเป็นคนละตัวกับ $A

ตัวอย่างต่อไปนี้เป็นการกำหนดค่าให้กับตัวแปรในลักษณะที่แตกต่างกัน

● การกำหนดค่าของตัวแปรที่เป็นตัวเลข โดยการใช้เครื่องหมายทางคณิตศาสตร์

$a = 1 + 2; คือการนำเอา 1 บวกกับ 2 แล้วเอาผลลัพธ์เก็บไว้ในตัวแปร $a

$a = 3 - 4; คือการนำเอา 4 ลบออกจาก 3 แล้วเอาผลลัพธ์เก็บไว้ในตัวแปร $a

$a = 5 * 6; คือการนำเอา 5 คูณกับ 6 แล้วเอาผลลัพธ์เก็บไว้ในตัวแปร $a

$a = 7 / 8; คือการนำเอา 8 ไปหาร 7 แล้วเอาผลลัพธ์เก็บไว้ในตัวแปร $a

$a = 9 ** 10; คือการนำเอาค่าของ 9 ยกกำลัง 10 เก็บไว้ในตัวแปร $a

$a = 5 % 2; คือการนำเอาเศษจากการหาร 5 ด้วย 2 เก็บไว้ในตัวแปร $a

++$a; คือการเพิ่มค่าของ $a เข้าไปอีก 1 แล้วเอาค่านี้เก็บไว้ในตัวแปร $a

$a++; คือการเก็บค่าปัจจุบันของ $a ไว้ในตัวแปร $a แล้วจึงเพิ่มค่าของ $a เข้าไปอีก 1

- -$a; คือการลดค่าของ $a ลงไป 1 แล้วเอาค่านี้เก็บไว้ในตัวแปร $a

$a- -; คือการเก็บค่าปัจจุบันของ $a ไว้ในตัวแปร $a แล้วจึงลดค่าของ $a ลงไป 1

● การกำหนดค่าของตัวแปรชนิดสตริง

$a = $b . $c คือการนำเอาสตริงของตัวแปร $c ต่อท้ายสตริงของตัวแปร $b แล้วเก็บค่านี้ไว้ใน $a

$a = $b x $c คือการนำเอาสตริงของตัวแปร $b มาต่อเข้าด้วยกัน $c ครั้งแล้วเก็บค่านี้ไว้ใน $a

● การกำหนดค่าให้กับตัวแปรในแบบอื่น ๆ

$a = $b คือการนำค่าในตัวแปร $b เก็บไว้ในตัวแปร $a

$a += $b คือการบวกค่าที่อยู่ใน $b เข้ากับค่าใน $a แล้วเก็บค่านี้ไว้ในตัวแปร $a

$a -= $b คือการบวกลบที่อยู่ใน $a ด้วยค่าที่อยู่ใน $b แล้วเก็บค่านี้ไว้ในตัวแปร $a

$a .= $b คือการนำเอาค่า $b ต่อท้ายค่าใน $a แล้วเก็บค่านี้ไว้ในตัวแปร $a

ครับ... และที่ผ่านมาก็คือการกำหนดค่าให้กับตัวแปรในแบบต่าง ๆ คราวนี้เราลองมาดูตัวอย่างการใช้งานตัวแปรในโปรแกรมง่าย ๆ ที่พัฒนามาจากโปรแกรม HELLO WORLD ดังต่อไปนี้

```
#!/usr/var/local/bin/perl
$a = "HELLO WORLD";
print $a;    # Print content of $a
```

หรืออีกแบบหนึ่งที่เราสามารถทำได้คือ

```
#!/usr/var/local/bin/perl
$a = "HELLO";
$b = "WORLD";
print $a .    . $b;
```

นอกจากนั้นเราอาจใช้สัญลักษณ์อื่น ๆ มาช่วยในเรื่องของการแสดงผลให้มีประสิทธิภาพมากขึ้นได้ เช่น

\n คือการกำหนดให้มีการขึ้นบรรทัดใหม่

\p คือการกำหนดให้มีการเว้นระยะห่างเป็นระยะ 1 แท็บ

2. ARRAY VARIABLES คือตัวแปรที่เก็บค่ารายการของตัวแปรชนิด SCALARS นั่นคือ ตัวแปรชนิด SCALAR 1 ตัวเก็บค่าได้เพียงแค่หนึ่งค่า คืออาจจะเป็นตัวเลขหรือสตริงก็ได้ แต่ตัวแปรชนิด ARRAY หนึ่งตัวจะสามารถเก็บค่าชนิดเดียวกันได้มากกว่าหนึ่งค่า เช่น

```
@food = ("apples" , "pears" , "eels");
@music = ("whistle" , "flute")
$food[2]
```

ในบรรทัดแรกเป็นการกำหนดตัวแปรชนิด array ชื่อว่า @food ซึ่งมีข้อมูลเก็บอยู่สามตัวคือ "apples", "pears" และ "eels" ส่วนบรรทัดที่สองเป็นการกำหนดตัวแปรชื่อ @music มีชนิดเป็น array เหมือนกัน แต่มีข้อมูลเก็บอยู่เพียงแค่สองตัวคือ "whistle" และ "flute" ข้อสังเกตก็คือคุณจะเห็นว่าในการกำหนดตัวแปรชนิด array จะมีข้อแตกต่างจากตัวแปรชนิด SCALAR ตรงที่สัญลักษณ์หน้าตัวแปร ถ้าเป็น SCALAR จะใช้เครื่องหมาย $ นำหน้า แต่ถ้าเป็น ARRAY จะใช้สัญลักษณ์ @ นำหน้า

ทีนี้ในการอ้างถึงข้อมูลของตัวแปรชนิด ARRAY นั้นเราสามารถทำได้โดยการอ้างชื่อของตัวแปรชนิด ARRAY นั้นแล้วตามด้วยลำดับที่ของค่าที่เราต้องการ ซึ่งการนับลำดับที่นั้นให้เริ่มนับจากค่าที่อยู่ทางซ้ายสุดเป็นต้นไป โดยลำดับที่นั้นจะเริ่มจาก 0 เช่น ในบรรทัดที่สาม $food[2] เป็นการอ้างถึงข้อมูลตัวที่สามตัวแปร array ที่ชื่อว่า @food ซึ่งค่านั้นก็คือ "eels" และที่น่าสังเกตอีกอย่างหนึ่งคือ เครื่องหมายที่เราใช้นำหน้าตัวแปรที่ต้องการอ้างถึงข้อมูลที่เราต้องการนั้นคือเครื่องหมาย $ แทนที่จะเป็นเครื่องหมาย @ ทั้งนี้เพราะข้อมูลแต่ละตัวที่ ARRAY @food เก็บไว้นั้นคือตัวแปรชนิด SCALAR

จากเรื่องราวของการกำหนดตัวแปรชนิด ARRAY และการอ้างอิงถึงข้อมูลแต่ละตัวที่อยู่ใน ARRAY แล้ว สำหรับเรื่องที่ผมจะกล่าวถึงต่อไปก็คือเรื่องของการกำหนดค่าให้กับตัวแปร ARRAY ที่เราสร้างขึ้นมา ซึ่งวิธีการให้ค่าตัวแปร ARRAY นี้ก็มีวิธีที่เหมือน ๆ กันกับการให้ค่าตัวแปรชนิด SCALAR ดังตัวอย่างต่อไปนี้

```
@moremusic = ("organ" , "whistle" , "flute" , "harp");
@moremusic = ("organ" , @music , "harp");
```

ในบรรทัดแรกเป็นการกำหนดค่าให้กับตัวแปร ARRAY ชื่อ @moremusic ซึ่งเป็น ARRAY ที่สามารถเก็บข้อมูลได้สี่ตัว ซึ่งค่าที่กำหนดให้กับตัวแปร @moremusic ในบรรทัดแรกนี้ เป็นค่าคงที่ทั้งสี่ตัว ในขณะที่ในบรรทัดที่สองเป็นการกำหนดค่าให้กับตัวแปร @moremusic เหมือนกับในบรรทัดแรก แต่แตกต่างกันที่ในการกำหนดค่าให้กับตัวแปร ARRAY นั้น คุณจะเห็นว่าเราสามารถนำค่าจากตัวแปรอื่น ๆ มากำหนดให้กับตัวแปร @moremusic ได้ นั่นคือเราสามารถใช้ตัวแปร @music ซึ่งเป็นตัวแปร ARRAY ที่เก็บข้อมูลไว้สองตัวคือ "whistle" และ "flute" มาใช้กำหนดค่าให้กับตัวแปร @moremusic แทนการใช้ค่าคงที่ดังตัวอย่างในบรรทัดแรกก็ได้นะครับ แถมยังเป็นการดีซะอีกคือทำให้ประโยคที่เราเขียนมีขนาดกะทัดรัด สามารถตรวจสอบได้ง่าย และตัวอย่างต่อไปจะเป็นตัวอย่างการกำหนดค่าให้กับตัวแปร ARRAY เช่นกัน แต่คราวนี้จะเป็นการเพิ่มข้อมูลเข้าไปต่อท้ายข้อมูลตัวสุดท้ายของตัวแปร ARRAY ตัวนั้น ซึ่งคราวนี้เราจะใช้คำสั่ง push แทนการใช้เครื่องหมาย assignment(=) ดังต่อไปนี้

```
push(@food , "eggs");
push(@food , "eggs" , "lard");
push(@food , @morefood)
```

ในบรรทัดแรกเป็นการเพิ่มค่า "eggs" เข้าไปต่อท้าย ARRAY @food และในบรรทัดที่สองและสามก็เช่นเดียวกัน แต่เป็นการเพิ่มข้อมูลมากกว่าหนึ่งตัวเข้าไปใน ARRAY ทีนี้เมื่อมีการเพิ่มค่าเข้าไปใน ARRAY แล้ว ความสามารถอีกอย่างหนึ่งที่เราทำได้ก็คือ การลบข้อมูลออกจาก ARRAY ทำได้โดยการใช้คำสั่ง pop ดังนี้

$grup = pop(@food); คือการลบข้อมูลตัวสุดท้ายออกจาก ARRAY @food และเอาค่านี้เก็บไว้ในตัวแปร $grup ที่มีชนิดเป็น SCALAR

นอกจากนั้นเราสามารถทำงานกับตัวแปร ARRAY ได้อีกมากมายหลายแบบ เช่น

$f = @food; เป็นการนำค่าความยาวของ ARRAY @food (จำนวนข้อมูลที่สามารถเก็บได้) ไปเก็บไว้ที่ตัวแปร $f

$f = "@food"; เป็นการนำค่าทั้งที่อยู่ในตัวแปร ARRAY @food มาเรียงต่อกันเป็นสตริง โดยจะมีช่องว่างคั่นระหว่างข้อมูลแต่ละตัว แล้วนำค่านี้ให้กับตัวแปร $f ซึ่งเป็น SCALAR

($a , $b) = ($c , $d); เหมือนกับการกำหนดค่าโดยการใช้ประโยค $a = $c และ $b = $d

($a , $b) = @food; คือการนำข้อมูลสองตัวแรกไปกำหนดให้กับตัวแปร $a และ $b

($a , @somefood) = @food; คือการนำข้อมูลตัวแรกไปกำหนดให้กับตัวแปร $a และ นำข้อมูลตัวถัด ๆ ไปกำหนดให้กับตัวแปร @somefood ซึ่งขึ้นอยู่กับว่า @somefood สามารถเก็บข้อมูลได้กี่ตัว

(@somefood , $a) = @food; คือการนำข้อมูลที่อยู่ใน @food ไปกำหนดให้กับตัวแปร @somefood ส่วนตัวแปร $a นั้นไม่ได้มีการกำหนดค่าให้

$#food คือการหาตำแหน่งของข้อมูลตัวสุดท้ายในตัวแปร @food

ส่วนเรื่องของการแสดงผลหรือการแสดงข้อมูลที่อยู่ใน ARRAY ออกมา เราสามารถทำได้โดยการใช้คำสั่ง print เหมือนกับที่ใช้ในการแสดงข้อมูลแบบ SCALAR เช่น print @food; เป็นการแสดงข้อมูลที่เก็บไว้ในตัวแปร @food ออกมาที่หน้าจอ

บทสรุปก่อนพบกันใหม่

ครับ... สำหรับคราวนี้คุณคงพอที่จะเห็นภาพคร่าว ๆ ของการเขียนโปรแกรมโดยการใช้ PERL กันแล้วนะครับ ซึ่งคุณจะเห็นว่าการเขียนโปรแกรมนั้นทำได้ไม่ยากอย่างที่ผมบอกไว้ และสำหรับคราวหน้าผมจะมาเล่าเรื่องราวของการเขียน PERL ที่ลึก ๆ ขึ้นไปอีกสักหน่อยนะครับ แต่รับรองว่าไม่ยากเกินความสามารถคุณแน่

ค่ายเยาวชนสมองแก้ว

มหาวิทยาลัยเกษตรศาสตร์ จัดค่ายเยาวชนสมองแก้วเป็นประจำทุกปี ครั้งนี้เป็นครั้งที่ 10 ระหว่างวันที่ 19-25 เมษายน 2540 ที่มหาวิทยาลัยเกษตรศาสตร์ วิทยาเขตกำแพงแสน

โดยการจัดค่ายอบรมในครั้งนี้ มีจุดมุ่งหมายในการเสริมความรู้ให้กับเยาวชนที่สนใจในเรื่องวิทยาศาสตร์ เทคโนโลยีและคอมพิวเตอร์ ซึ่งกิจกรรมจะเน้นให้เยาวชนสนุกกับคอมพิวเตอร์ การสร้างสรรสิ่งประดิษฐ์จากอิเล็กทรอนิกส์ ตลอดจนกิจกรรมที่สร้างสรรค์ทางด้านวิทยาศาสตร์อีกมากมาย

ผู้ที่สนใจสามารถติดต่อสมัครได้ที่ ภาควิชาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยเกษตรศาสตร์ เขตบางเขน โทรศัพท์ 562-0951-7 แฟกซ์ 562-0950